PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2003-296302
(43)Date of publication of application : 17.10.2003

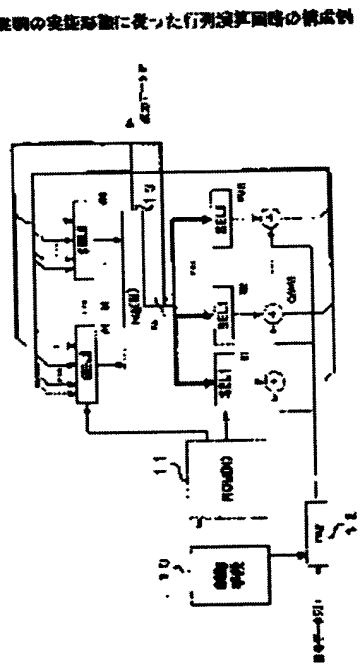
(51)Int.Cl. G06F 17/16
H03M 13/09
H03M 13/19

(21)Application number : 2002-098025
(22)Date of filing : 29.03.2002

(71)Applicant : FUJITSU LTD
(72)Inventor : OTA MITSUHIKO

(54) MATRIX ARITHMETIC PROCESSOR

(57)Abstract:
PROBLEM TO BE SOLVED: To provide a matrix arithmetic processor capable of quickly executing a matrix arithmetic operation in a small circuit scale.
SOLUTION: An input signal data column I is temporarily stored in a reg 12, and inputted to an adder in response to an instruction from a control means 10. The control means 10 instructs an ROM 11 in which a check matrix H is stored to obtain a position where '1' is erected in a certain column of the check matrix. The ROM 11 instructs selectors SEL1 #1 to #CW to select values corresponding to the position of 1 in the check matrix from among values from a reg(M) 13, and to transmit the values to the adder. The added values are selected by selectors SEL2 instructed by the ROM 11, and inputted to the reg(M) 13. As for non-added values, the output values of the reg(M) 13 are inputted as they are, or inputted through the selectors SEL2 to the reg(M). This process is repeated until all the arithmetic operations end.



LEGAL STATUS

[Date of request for examination] 10.12.2004

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号
特開2003－296302
（P2003－296302A）
(43)公開日　平成15年10月17日(2003.10.17)

(51)Int.Cl.'	識別記号	ＦＩ	テーマコード'(参考)
Ｇ０６Ｆ　17/16		Ｇ０６Ｆ　17/16	Ｍ　５Ｂ０５６
			Ｐ　５Ｊ０６５
Ｈ０３Ｍ　13/09		Ｈ０３Ｍ　13/09	
13/19		13/19	

審査請求　未請求　請求項の数５　ＯＬ　（全９頁）

(21)出願番号　特願2002－98025(P2002－98025)

(22)出願日　平成14年３月29日(2002.3.29)

(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号

(72)発明者　太田　光彦
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

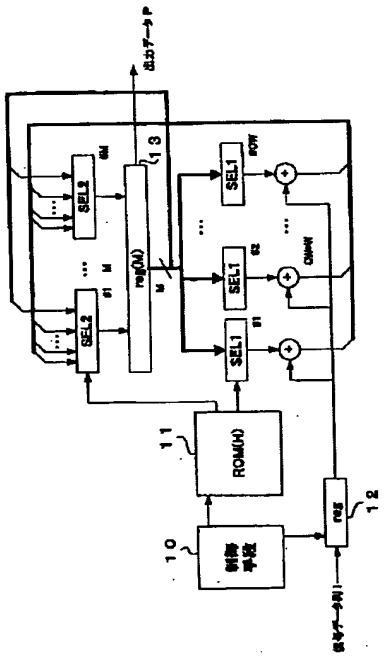
(74)代理人　100074099
弁理士　大菅　義之　（外１名）

Ｆターム(参考)　5B056 AA06 BB39 BB42 BB72 GG01 HH03
5J065 AA01 AB01 AC03 AD01 AH01 AH05 AH06 AH09

(54)【発明の名称】　行列演算処理装置

(57)【要約】
【課題】小さな回路規模で高速に行列演算することのできる行列演算処理装置を提供する。
【解決手段】入力信号データ列Ｉを一旦reg１２に格納し、制御手段１０の指示により、加算器に入力する。制御手段１０は検査行列Ｈが格納されているＲＯＭ１１に指示し、検査行列のある列において、“１”が立っている位置を取得する。ＲＯＭ１１は、ＳＥＬ１＃１～＃ＣＷに指示して、reg（Ｍ）１３からの値の内、検査行列が１の位置に対応する値をセレクタＳＥＬ１で選択し、加算器に送る。加算された値は、やはりＲＯＭ１１から指示を受けたセレクタＳＥＬ２によって選択され、reg（Ｍ）１３に入力される。加算されなかった値は、reg（Ｍ）１３の出力値がそのまま、また、セレクタＳＥＬ２を介してreg（Ｍ）に入力される。これを全ての演算が終わるまで繰り返す。

本発明の実施形態に従った行列演算回路の構成例



1

【特許請求の範囲】

【請求項１】行列の行列要素を格納する格納手段と、初期値として全て０である値を格納し、順次行われる演算結果を順次格納するレジスタ手段と、該レジスタ手段から出力される値と入力データ値とを加算する加算手段と、前記行列の行列要素値に基づいて、必要な値を該レジスタ手段から該加算手段に入力し、入力データ値と該レジスタ手段からの値を加算させる演算制御手段と、加算器の出力と、該レジスタ手段の出力を適切に選択して再び該レジスタ手段に格納するループバック手段と、を備える行列演算処理装置。

【請求項２】前記格納手段は、検査行列の行列要素が１となる行列要素の位置のみを情報として格納することを特徴とする請求項１に記載の行列演算処理装置。

【請求項３】前記演算制御手段は、入力データ値の入力順に合わせて前記格納手段から必要な情報を出力させることを特徴とする請求項１に記載の行列演算処理装置。

【請求項４】前記行列は、データの符号化に伴う誤り訂正符号の検査行列であることを特徴とする請求項１に記載の行列演算処理装置。

【請求項５】前記行列は、ＬＤＰＣ（Low Density Parity Check）符号の検査行列であることを特徴とする請求項１に記載の行列演算処理装置。

2

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、行列演算処理装置に関する。

【０００２】

【従来の技術】現在、パソコンをはじめ多くの情報処理装置が実用化され、実際に使用されている。このような情報処理装置においては、データを格納したり、転送する場合、データを符号化して格納あるいは転送する。特に、デジタル信号を磁気ディスク、光ディスク、光磁気ディスクなどの記録再生媒体あるいはネットワークから受信して復号する場合に使用する誤り訂正としてＬＤＰＣ（Low Density Parity Check）符号がある。

【０００３】エラー訂正などを行う場合、Ｎビットの信号データ列ＩとＮ×Ｍの検査行列Ｈを用いて処理結果Ｐを得るには、$P=H*I^{T}$という行列演算が必要となる。例えば、

【０００４】

【数１】

$$I=[i_0,i_1,\Lambda,i_{N-1}],P=[p_0,p_1,\Lambda,p_{M-1}],H=\begin{bmatrix} h_{00} & h_{01} & \Lambda & h_{0N-1} \\ M & M & M & M \\ h_{M-10} & h_{M-11} & \Lambda & h_{M-1N-1} \end{bmatrix}$$

【０００５】とした場合、

【０００６】

【数２】

$$P_m=\sum_{n=0}^{N-1} h_{mn} * i_n \quad \cdots\cdots\cdots(1)$$

【０００７】となる。例えば、記録再生装置の代表例である磁気ディスク装置に関して、エラー訂正機能が搭載され、その符号の一つとしてＬＤＰＣ符号があり、この符号にも行列演算が必須となっている。

【０００８】ここで、パリティ計算やＬＤＰＣ復号で用いられる検査行列は通常、行列要素が１あるいは０であるので、行列中要素が１である項を加算することになり、（１）式は、

【０００９】

【数３】

$$P_m=\sum_{n=0/h_{mn}=1}^{N-1} i_n \quad \cdots\cdots\cdots(2)$$

【００１０】となる。

【００１１】

【発明が解決しようとする課題】図７は、従来の行列演算回路の例である。従来、上記の処理を行い処理結果Ｐを得るためには、信号データ列Ｉの全データを取得後、（２）式の計算を行う必要がある。

【００１２】信号データ列Ｉは、レジスタ４０にその全データが蓄積された後、行列データＨを格納したＲＯＭ４１などから読み出された各行の中の１となっている項がセレクタＳＥＬ４２によってセレクトされ、加算器４６で加算される。加算結果は、レジスタ４３に格納される。ここでＲＷは各行の１の最大個数とする。これをＭ回繰り返すことで処理結果Ｐを得る。ここではＰの全データを得るまでセレクタ／加算器を共用した場合でも、Ｎ＋Ｍ時間が必要となり遅延が大きいという問題がある。また、回路規模も大きな保持レジスタＮが必要であったり、ＲＷ個の加算器が必要となる。

【００１３】図８は、別の従来例を示す図である。本従来例では、信号データ列Ｉはレジスタ４４にその全データが蓄積された後、行列Ｈに従って結線された加算器４７により処理結果Ｐの全データを計算する。結果はレジスタ４５に格納され、出力される。この場合、Ｐの全データを得るまで必要な時間はＮであるが、保持レジスタＮが大きいと言うことと、ＲＷ×Ｍ個の加算器が必要となり回路規模が大きいという問題がある。

【００１４】本発明の課題は、小さな回路規模で高速に行列演算することのできる行列演算処理装置を提供することである。

【００１５】

【課題を解決するための手段】本発明の行列演算処理装置は、行列の行列要素を格納する格納手段と、初期値として全て０である値を格納し、順次行われる演算結果を順次格納するレジスタ手段と、該レジスタ手段から出力される値と入力データ値とを加算する加算手段と、前記行列の行列要素値に基づいて、必要な値を該レジスタ手段から該加算手段に入力し、入力データ値と該レジスタ手段からの値を加算させる演算制御手段と、加算器の出力と、該レジスタ手段の出力を適切に選択して再び該レジスタ手段に格納するループバック手段とを備える。

【００１６】本発明によれば、従来に比べ、加算器が少なく、回路規模が小さくなる。また、行列の格納も必要な情報のみを格納するようにすれば、メモリ容量も小さくでき、小型かつ高速な行列演算処理装置を提供することができる。

【００１７】

【発明の実施の形態】Ｎビットの信号データ列ＩとＮ×Ｍの検査行列Ｈを用いて処理結果Ｐを得るＰ＝Ｈ×I^Tという行列演算において、検査行列Ｈの各列単位で必要となる演算を行い、処理結果Ｐの各要素を行分累積加算することで処理遅延／回路規模を削減する。特に、符号化における誤り訂正符号の検査行列においては、検査行列の列数Ｍが行数Ｎよりもかなり小さくなるので、列方向に並列に計算し、行方向に累積演算するようにすることにより、加算器の数を減らし、回路規模を小さくすることができる。

【００１８】レジスタなどの処理結果Ｐの保持手段、ＲＯＭなどの検査行列Ｈの保持手段、アドレスカウンタなどの信号データ列Ｉの受信により検査行列Ｈ及び処理結果Ｐの読み出し、格納を制御する手段、加算器などの演算手段を備え、信号データ列Ｉの受信毎に入力データの処理が必要となる列データをＨから得て、対象となる処理結果Ｐの必要項を読み出し、受信データと演算、結果を処理結果Ｐの保持手段に書き戻す。これを信号データ列Ｉの全受信データに繰り返し行うことで、処理結果Ｐを得る。

【００１９】図１は、本発明の実施形態に従った行列演算回路の構成例である。制御手段１０は信号データ列Ｉを受信制御し、保持手段reg１２に保持する。また、受信データ位置から検査行列Ｈの保持手段１１より列の処理対象の処理結果Ｐの位置を得る。処理結果Ｐの格納手段reg（Ｍ）１３はデータ受信前に全て０に初期化される。制御手段１０と保持手段１１によって処理対象の処理結果ＰはデータセレクタＳＥＬ１で選択され、入力データと加算処理される。ここで加算器は列中の１の最大個数ＣＷ個。ＳＥＬ１はＭ→ＣＷのセレクタでＣＷ個、ＳＥＬ２は（ＣＷ＋１）→１のセレクタでＭ個となる。加算結果及びreg（Ｍ）１３より読み出されたデータは

ＳＥＬ２により処理データか元データが選択されreg（Ｍ）１３に書き戻される。

【００２０】すなわち、図１においては、信号データ列Ｉはreg１２に１ビット格納され、制御手段１０によって呼び出され、各加算器に入れられる。制御手段は、ＲＯＭ１１の中から検査行列Ｈの適切な列の内、１となっている位置を検出し、reg（Ｍ）１３から来た信号の内、検査行列の当該列内の１となっている位置に対応するセレクタＳＥＬ１がreg（Ｍ）１３からの値を選択し、信号データ列と加算させる。加算結果は、セレクタＳＥＬ２に送られ、セレクタＳＥＬ２では、加算が行われたデータ値については、加算器からの値をreg（Ｍ）１３に入力し、加算が行われなかった値については、reg（Ｍ）１３からの値を再び、reg（Ｍ）１３に入力する。そして、全ての信号データ列Ｉの受信、演算が終わった時点でreg（Ｍ）１３は、出力データＰを出力する。

【００２１】より概念的に述べれば、行列の縦方向を列方向、横方向を行方向とすると、入力信号データ列Ｉを読み込んだとき、Ｉの入力順を予め知っておくことによって、Ｉの要素をどの列に乗算すべきかを決定しておく。そして、Ｉが入力されると、演算すべき検査行列の列を読み出し、並列に列方向の演算をする（実際には、検査行列の要素は０か１であるので、行列要素が１である行の値をそのまま加算器に入力する）。そして、演算結果をreg（Ｍ）に格納し、Ｉが順次入力される毎に、行方向に加算していき、全ての加算が終わった時点でパリティの列ベクトルが得られ、演算終了とする手順である。

【００２２】本装置によればデータの処理遅延はＮ、回路規模は加算器はＣＷ個、保持レジスタはＭとなる。検査行列Ｈの保持手段の例として、列中の１の位置のアドレス保持する場合、（Cadd1、Cadd2、Cadd3）をＲＯＭ（Ｈ）の１番目のアドレスに格納する。ここでCaddnは列ｎ番目の１の位置を表し、１の個数がＣＷ個未満の場合は０とする。

【００２３】例えば、Ｈ（Ｎ＝８、Ｍ＝５）を、

【００２４】

【数４】

$$H=\begin{pmatrix}1&0&1&1&0&1&1&0\\0&1&0&1&1&1&1&1\\1&0&1&0&0&0&1&1\\1&1&0&1&1&0&0&1\\0&1&1&0&1&1&0&0\end{pmatrix}$$

【００２５】とした場合は、

【００２６】

【表１】

アドレス	データ
0	(1、3、4)
1	(2、4、5)
2	(1、3、5)
3	(1、2、4)
4	(2、4、5)
5	(1、2、5)
6	(1、2、3)
7	(2、3、4)

【００２７】となる。この場合各アドレスは３ビット（０～５＜８）で表現可能なので、ＣＷ＝３なら３×３＝９ビットを１ワードとして格納すれば良い。入力データ列Ｉに応じ、０番目なら（１、３、４）が出力され、各ＳＥＬ１は左から１、３、４のＭを選択し、ＳＥＬ２は左から１、３、４番目のＳＥＬ２が各ＳＥＬ１からの信号を、それ以外のＳＥＬ２はＭからの信号を選択する。

【００２８】ＳＥＬ１、ＳＥＬ２の制御実現例を以下に示す。ＳＥＬ１はＭ→１のセレクタで、制御信号がｍ（整数）を示すものであるとすると、ｍ番目のデータを選択出力し、この場合ＲＯＭからの出力をＭＳＢより３ビットずつ制御信号ｍとすれば良い。

【００２９】ＳＥＬ２は４（ＣＷ＋Ｍ）→１のセレクタで、制御信号は、０→Ｍ、１→ＳＥＬ１＃１、２→ＳＥＬ１＃２、４→ＳＥＬ１＃ＣＷを選択するものとし、ＲＯＭからの出力を表２のデータマルチプレクサデコーダに入力、各デコーダの出力を図Ａのように制御信号とすることで実現できる。

【００３０】

【表２】

入力	出力データ
0	0（00000b）
1	1（00001b）
2	2（00010b）
3	4（00100b）
4	8（01000b）
5	16（10000b）

【００３１】図２は、本発明の実施形態の動作を示す回路例である。制御信号は、ＲＯＭ（Ｈ）１から出力され、セレクタＳＥＬ１＃１～ＳＥＬ１＃ＣＷ（２－１～２－ＣＷ）に入力されると共に、マルチプレクサデコーダＤＥＣ３－１～３－ｎそれぞれに入力される。マルチプレクサデコーダＤＥＣ３－１～３－ｎからは、ＲＯＭ（Ｈ）１からの制御信号をセレクタＳＥＬ２の選択信号にデコードした信号が出力される。ＳＥＬ２は、これらマルチプレクサデコーダＤＥＣ３－１～３－ｎからの信号に基づいて制御される。Ｈ行列の読み出しアドレスの発生方法の例図３～図６は、本発明の実施形態に従った制御手段の実現例を示す図である。

【００３２】まず、図３において、入力データ列Ｉの先頭においてはdata_startが、また、データＩが有効な期間はdata_enableが図３のような場合、カウントイネーブル（“１”でカウントＵＰ）、クリア（“１”で全て０）付きのカウンタ２２を用いることで制御手段を構成可能である。ここで、前項の行列の場合は３ビットカウンタ（０～７）で可能である。

【００３３】また、Ｍの保持レジスタは図３のようにdata_enableが“１”の場合はＳＥＬ２出力を、“０”の場合はデータをセレクタ２０で選択し、ＦＦ２１に保持することで実現できる。

【００３４】ここで、data_enableは図４のように不連続でも良い。入力データＩが昇順で入力されない場合、降順であればＨの格納を逆順で格納することで実現できる。

【００３５】また、図４にてカウンタをロード機能付きダウンカウンタに変更、data_startにて値Ｎ－１をロードし、data_enableでダウンカウントすることでも実現可能である。

【００３６】入力データＩが予め定義された順番の場合は、Ｈの格納をその定義順で行うことで実現可能である。入力データＩがインタリーブされた場合はＨの格納をその順番で格納することで、実現可能である。

【００３７】また、図５の構成でも可能である。ｎはインタリーブの間隔とする。ここでｎ＝２とすると入力順は前項の例（０～７）では０、２、４、６、１、３、

５、７となる。

【００３８】ＲＥＧ３３はアドレスを保持するＦＦ、各セレクタ３１、３２はdata_enableが“１”の時、図５においてセレクタ３１、３２の下側に入っているデータをセレクトする。比較器３４は、カウンタ３０のカウントイネーブルと、セレクタ３１、３２の上側のポートへの値の入力を行うものである。

【００３９】各動作は、

１．data_startでＲＥＧを全て０、カウンタ３０はクリアで“１”をロードする。

２．data_enableでＲＥＧ３３にｎを加算した値をロード（設定）。

３．ＲＥＧ＞Ｎ－１を比較、真なら４へ、偽なら２へ

４．data_startなら１へ、それ以外はdata_enableでＲＥＧ３３へカウンタの値を設定、カウンタ３０はＣＥで＋１し、２へとなる。

【００４０】本構成でｎ＝１とすればインタリーブ無しの昇順構成となる。また、図６の構成にし、

１．data_startでＲＥＧ３３をＮ－１、カウンタ３０はクリアでＮ－２をロードする。

２．data_enableでＲＥＧ３３にｎを減算した値をロード（設定）。

３．ＲＥＧ＝０を比較、真なら４へ、偽なら２へ

４．data_startなら１へ、それ以外はdata_enableでＲＥＧ３３へカウンタの値を設定、カウンタ３０はＣＥで－１し２へ

とすることで逆順のインタリーブ構成となる。

【００４１】以上のような実施形態においては、ＬＤＰＣ符号における検査行列の一例で比較した場合、Ｎ＝４３５２、Ｍ＝２５６、ＣＷ＝３、ＲＷ＝５１とすると、本実施形態では、従来例に比較して保持メモリでＭ／Ｍ＝１／４、加算器は図７の構成と比較してＣＷ／ＲＷ＝３／５１、図８の構成と比較して、ＣＷ／ＲＷ×Ｍ＝３／１３０５６となる。

【００４２】本実施形態の装置では処理は各列単位で行われる。これによりＮビットの信号データ列Ｉ受信順序によらず、処理を行うことが可能となる。また、各列単位で処理を行うことでＮビットの信号データ列Ｉの任意の位置から処理を行うことが可能となる。

【００４３】更に、本実施形態においては検査行列Ｈの保持を処理が必要となる項のアドレスのみを保持することで保持手段の回路規模が削減可能となる。行列をそのまま保持した場合は、Ｍビット×Ｎアドレス分の容量となるが、アドレスで保持した場合、$\log_2$（Ｍ）ビット×ＣＷの容量となる。

【００４４】また、信号データ列Ｉの受信順序の保存を行列保持手段で行うことで、受信順序に依存しない処理装置を構成可能となる。また、上記で説明したように信号データ列Ｉ受信順序が逆順で送られる場合、行列を逆アドレスで保持することで、回路を変更することなく構成可能となる。

【００４５】上記装置において、信号データ列Ｉの受信順序の保存を行列保持手段に与える制御データで行うことで受信順序に依存しない処理装置を構成可能となる。例えば、受信順序アドレスをレジスタによるテーブルとして構成することにより、受信順序に非依存及び動的な変更が可能となる。

【００４６】上記実施形態においては、回路規模の比較的小さいセレクタの数が増えるが、従来技術に比べ、回路規模の比較的大きい加算器の数が減るので、全体としては、回路規模を小さく抑えることができる。

【００４７】また、ＲＯＭは、検査行列の行列要素が１である部分の行列内での位置のみを格納すればよいので、行列要素を全部記憶する必要が無く、メモリ容量が小さくて済む。更に、ＬＤＰＣの場合には、行列要素の内、１の数が比較的少ないので、特にメモリ容量の節約に有効である。

【００４８】なお、上記実施形態の説明においては、入力データ列Ｉは、信号値が２値のビット列であることを前提にしたが、必ずしもこれに限定されるものではなく、信号値が各要素の値として実数を取るような場合にも同様に適用可能である。

【００４９】ＬＤＰＣとその符号化については、詳しくは、下記文献を参考にされたい。・和田山正　「低密度パリティ検査符号とその復号方法について」磁気記録研究会発表論文、２００１年１２月、<http://vega.c.oka-pu.ac.jp/~wadayama/welcome_j.html>

・Tadashi Wadayama, "An Extention ofGallager Ensemble of Low Density Parity Check Codes", IEICE TRANS.FUNDAMENTALS, VOL. E85-A, NO.1 JANUARY 2002

（付記１）行列の行列要素を格納する格納手段と、初期値として全て０である値を格納し、順次行われる演算結果を順次格納するレジスタ手段と、該レジスタ手段から出力される値と入力データ値とを加算する加算手段と、前記行列の行列要素値に基づいて、必要な値を該レジスタ手段から該加算手段に入力し、入力データ値と該レジスタ手段からの値を加算させる演算制御手段と、加算器の出力と、該レジスタ手段の出力を適切に選択して再び該レジスタ手段に格納するループバック手段と、を備える行列演算処理装置。

【００５０】（付記２）前記格納手段は、検査行列の行列要素が１となる行列要素の位置のみを情報として格納することを特徴とする付記１に記載の行列演算処理装置。

（付記３）前記演算制御手段は、入力データ値の入力順に合わせて前記格納手段から必要な情報を出力させることを特徴とする付記１に記載の行列演算処理装置。

【００５１】（付記４）前記行列は、データの符号化に伴う誤り訂正符号の検査行列であることを特徴とする付記１に記載の行列演算処理装置。

（付記５）前記行列は、ＬＤＰＣ（Low Density Parity Check）符号の検査行列であることを特徴とする付記１に記載の行列演算処理装置。

【００５２】（付記６）行列の行列要素を格納する格納ステップと、初期値として全て０である値を格納し、順次行われる演算結果を順次レジスタ手段に格納するレジスタステップと、該レジスタ手段の格納値と入力データ値とを加算する加算ステップと、前記行列の行列要素値に基づいて、必要な値を該レジスタ手段の格納値から該加算ステップに渡し、入力データ値と該格納値を加算させる演算制御ステップと、加算器の出力と、該格納値を適切に選択して再び該レジスタ手段に格納するループバックステップと、を備える行列演算処理方法。

【００５３】（付記７）前記格納ステップでは、検査行列の行列要素が１となる行列要素の位置のみを情報として格納することを特徴とする付記６に記載の行列演算処理方法。

【００５４】（付記８）前記演算制御ステップでは、入力データ値の入力順に合わせて前記格納手段から必要な情報を出力させることを特徴とする付記６に記載の行列演算処理方法。

【００５５】（付記９）前記行列は、データの符号化に伴う誤り訂正符号の検査行列であることを特徴とする付記６に記載の行列演算処理方法。

（付記１０）前記行列は、ＬＤＰＣ（Low Density Parity Check）符号の検査行列であることを特徴とする付記６に記載の行列演算処理方法。

【００５６】

【発明の効果】本発明により、行列演算回路の遅延、回路規模を削減することが可能となる。

【図面の簡単な説明】

【図１】本発明の実施形態に従った行列演算回路の構成例である。

【図２】本発明の実施形態の動作を示す回路例である。

【図３】本発明の実施形態に従った制御手段の実現例を示す図（その１）である。

【図４】本発明の実施形態に従った制御手段の実現例を示す図（その２）である。

【図５】本発明の実施形態に従った制御手段の実現例を示す図（その３）である。

【図６】本発明の実施形態に従った制御手段の実現例を示す図（その４）である。

【図７】従来の行列演算回路の例である。

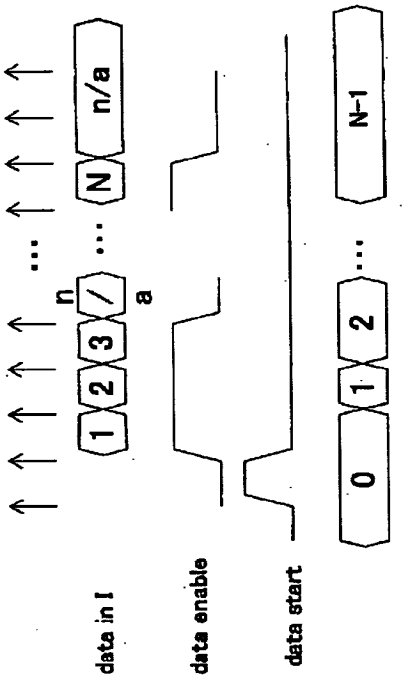
【図８】別の従来例を示す図である。

【符号の説明】

１０　制御手段
１１　ＲＯＭ
１２　reg
１３　reg（M）

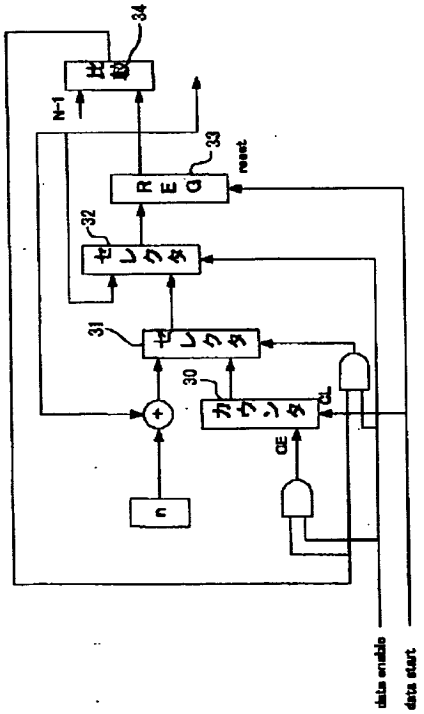
【図４】

本発明の実施形態に従った制御手段の実現例を示す図（その２）



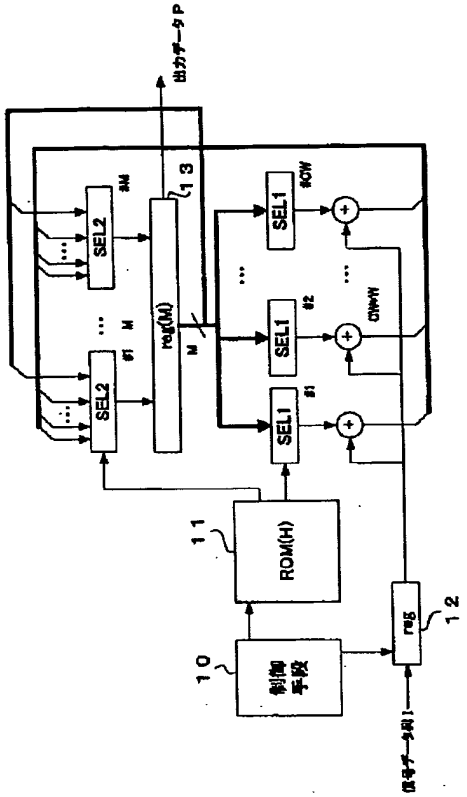
【図５】

本発明の実施形態に従った制御手段の実現例を示す図（その３）



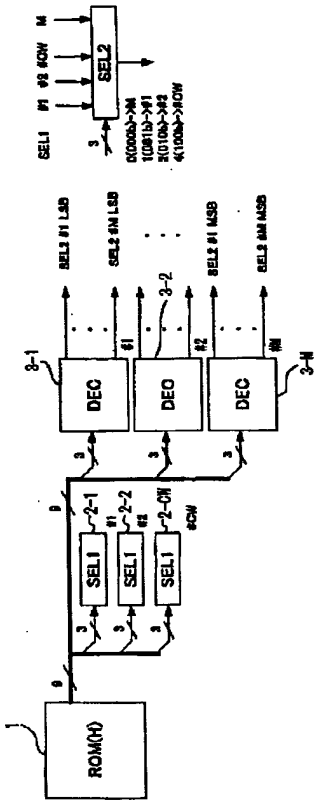
【図１】

本発明の実施形態に従った行列演算回路の構成例



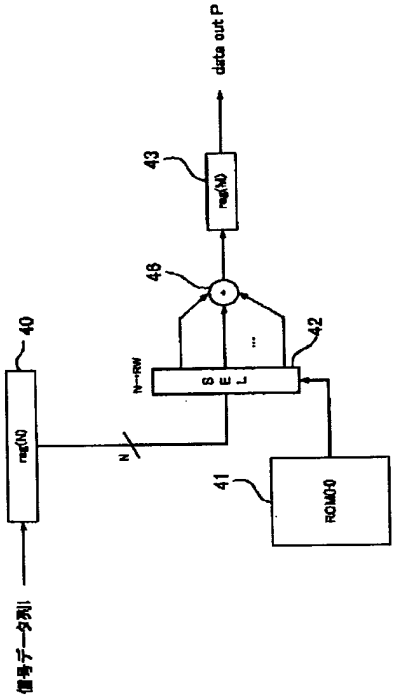
【図２】

本発明の実施形態の動作を示す回路例



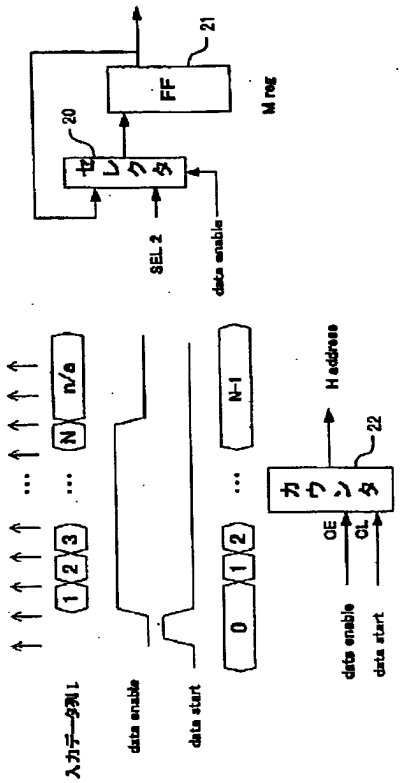
【図７】

従来の行列演算回路の例



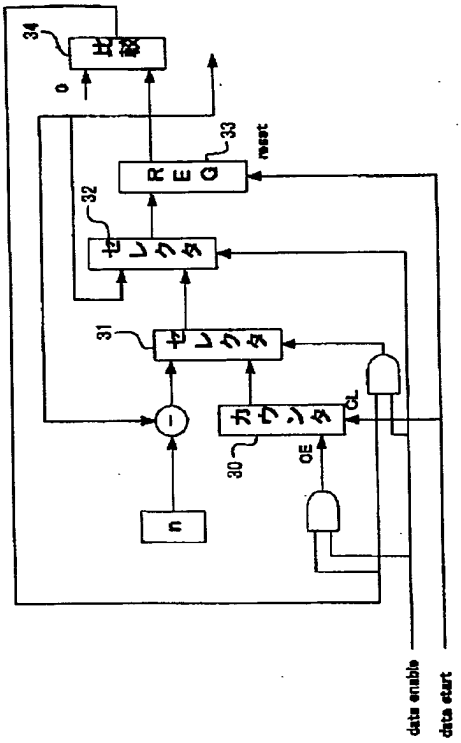
【図３】

本発明の実施形態に従った制御手段の実現例を示す図（その１）



【図６】

本発明の実施形態に従った制御手段の実現例を示す図（その４）



【図８】

別の従来例を示す図